II. AMENDMENTS

Please amend claim 55, as follows:

Claims:

1. (Original) A method for performing parallel decompression of compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, the method comprising:

receiving the compressed data, wherein the compressed data comprises tokens each describing one or more of the symbols in the uncompressed data;

examining a plurality of tokens from the compressed data in parallel in a current decompression cycle;

generating a plurality of selects in parallel in response to examining the plurality of tokens in parallel, wherein each of the plurality of selects points to a symbol in a combined history window; and

generating the uncompressed data comprising the plurality of symbols using the plurality of selects.

2. (Original) The method of claim 1, further comprising:

storing the uncompressed plurality of symbols from the current decompression cycle in the combined history window.

3. (Original) The method of claim 2,

wherein, in the current decompression cycle prior to said storing the uncompressed plurality of symbols, the combined history window includes an uncompressed plurality of symbols from any previous decompression cycles and zero or more data bytes from the current decompression cycle.

4. (Original) The method of claim 1, wherein said examining the plurality of tokens includes generating, for each token, size and count information and at least one of a data byte or index information; and

wherein said generating the plurality of selects in parallel uses the size and count information and at least one of the data byte or index information for each of the plurality of tokens.

5. (Original) The method of claim 4,

wherein a size for a token defines the number of bits comprising the token;

wherein a count for a token defines the number of symbols in the uncompressed data described by the token.

6. (Original) The method of claim 1,

wherein the combined history window includes one or more data bytes from the current decompression cycle; and

wherein one or more of the plurality of selects in the current decompression cycle point to one or more of the data bytes in the combined history window.

7. (Original) The method of claim 6, wherein said generating the plurality of selects in parallel uses index information generated for one or more of the plurality of tokens to generate the one or more selects pointing to the one or more of the data bytes in the combined history window.

8. (Original) The method of claim 1,

wherein the combined history window includes one or more uncompressed symbols from one or more previous decompression cycles; and

wherein one or more of the plurality of selects in the current decompression cycle point to one or more of the uncompressed symbols in the combined history window from the one or more previous decompression cycles.

9. (Original) The method of claim 8, wherein said generating the plurality of selects in parallel uses index information generated for one or more of the plurality of tokens to generate the one or more selects pointing to the one or more of the uncompressed symbols in the combined history window.

10. (Original) The method of claim 1, wherein the combined history window includes an uncompressed plurality of symbols from one or more previous decompression cycles and data bytes from the current decompression cycle, wherein said generating the plurality of selects in parallel comprises:

generating a first select to point to a data byte in the combined history window in response to a first token indicating that uncompressed data represented by the first token is the data byte; and

generating a second select to point to a first symbol in the combined history window in response to a second token indicating that uncompressed data represented by the second token includes the first symbol in the combined history window.

11. (Original) The method of claim 10, wherein the uncompressed data represented by the second token includes one or more symbols following the first symbol in the combined history window, wherein selects are generated to point to each of the one or more symbols in the combined history window comprising the uncompressed data represented by the second token.

12. (Original) The method of claim 1, wherein the combined history window includes an uncompressed plurality of symbols from one or more previous decompression cycles and data bytes from the current decompression cycle, wherein said generating the plurality of selects in parallel comprises:

generating a first select to point to a first symbol being decompressed from a first token in the current decompression cycle, wherein the first select is generated in response to a second token indicating that uncompressed data represented by the second token includes the first symbol, and wherein the first symbol is not in the combined history window.

13. (Original) The method of claim 12, further comprising resolving the first select to point to one of a symbol in the current combined history window or a data byte in the current combined history window.

14. (Original) The method of claim 12, further comprising copying a second select being generated for the first token to the first select, wherein the second select points to one of a symbol in the combined history window or a data byte in the combined history window.

15. (Original) The method of claim 1, wherein the combined history window includes an uncompressed plurality of symbols from one or more previous decompression cycles, wherein storing the uncompressed plurality of symbols from the current decompression cycle in the combined history window includes removing from the combined history window at least a portion of the uncompressed plurality of symbols from the one or more previous decompression cycles.

16. (Original) The method of claim 1, wherein said examining the plurality of tokens in parallel comprises:

extracting a portion of the compressed data as an input data, wherein the input data includes the plurality of tokens;

extracting one or more tokens from the input data; and

generating, for each token, size and count information and at least one of a data byte or index information.

17. (Original) The method of claim 16, wherein a plurality of decoders are operable to examine the plurality of tokens in parallel in the current decompression cycle, wherein said extracting the one or more tokens comprises:

a) determining if there are more tokens to be uncompressed in the input data;

b) determining if there is a decoder available in the plurality of decoders;

c) extracting a token from the input data to be a current token in response to determining that a decoder is available;

d) determining the number of uncompressed symbols to be generated by the current token;

e) assigning the current token to the available decoder; and

f) repeating a) through e) until a terminating condition is encountered.

18. (Original) The method of claim 17, wherein the terminating condition is encountered when:

step a) determines there are no more tokens in the input data; or

step b) determines there are no more decoders available in the plurality of decoders; or

step d) determines that a total number of uncompressed symbols to be generated from the plurality of tokens to be examined in parallel has met or exceeded a maximum output width.

19. (Original) The method of claim 17, wherein said extracting the one or more tokens further comprises determining a size of each of the one or more tokens.

20. (Original) The method of claim 17, wherein said extracting the one or more tokens further comprises:

determining if a token in the input data is a complete token or an incomplete token, wherein the token is extracted from the input data to be the current token in response to determining the token is a complete token;

wherein the terminating condition is encountered in response to determining the token is an incomplete token.

21. (Original) The method of claim 17, wherein said extracting the one or more tokens further comprises:

shifting the compressed data by a total size of one or more decompressed tokens, where shifting the compressed data prepares a next input data to be extracted.

22. (Original) The method of claim 1, wherein said generating a plurality of selects comprises generating at most M selects to at most M symbols in a decompression cycle.

23. (Original) The method of claim 22, wherein a token describes N symbols, and wherein completing the decompression of the token requires at least N/M decompression cycles, wherein N/M is rounded up to the next highest integer if N is not evenly divisible by M.

24. (Original) The method of claim 1, wherein said receiving the compressed data, said examining a plurality of tokens, said generating a plurality of selects, and said generating the uncompressed data comprising the plurality of symbols are performed substantially concurrently in a pipelined fashion.

25. (Original) A method for performing parallel decompression of compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, the method comprising:

receiving the compressed data, wherein the compressed data comprises tokens each describing one or more of the symbols in the uncompressed data;

examining one or more tokens from the compressed data in a current decompression cycle;

generating a plurality of selects in parallel in response to examining the one or more tokens, wherein each of the plurality of selects points to a symbol in a combined history window; and

generating the uncompressed data comprising the plurality of symbols using the plurality of selects.

26. (Original) The method of claim 25, further comprising:

storing the uncompressed plurality of symbols from the current decompression cycle in the combined history window.

27. (Original) The method of claim 26,

wherein, in the current decompression cycle prior to said storing the uncompressed plurality of symbols, the combined history window includes an uncompressed plurality of symbols from any previous decompression cycles and zero or more data bytes from the current decompression cycle.

28. (Original) The method of claim 25, wherein said examining the one or more tokens includes generating, for each token, size and count information and at least one of a data byte or index information; and

wherein said generating the plurality of selects in parallel uses the size and count information and at least one of the data byte or index information for each of the one or more tokens.

29. (Original) The method of claim 25, wherein said examining the one or more tokens comprises examining a plurality of tokens in parallel in the current decompression cycle.

30. (Original) The method of claim 25,

wherein the combined history window includes one or more data bytes from the current decompression cycle; and

wherein one or more of the plurality of selects in the current decompression cycle point to one or more of the data bytes in the combined history window.

31. (Original) The method of claim 30, wherein said generating the plurality of selects in parallel uses index information generated for the one or more tokens to generate the one or more selects pointing to the one or more of the data bytes in the combined history window.

32. (Original) The method of claim 25,

wherein the combined history window includes one or more uncompressed symbols from one or more previous decompression cycles; and

wherein one or more of the plurality of selects in the current decompression cycle point to one or more of the uncompressed symbols in the combined history window from the one or more previous decompression cycles.

33. (Original) The method of claim 32, wherein said generating the plurality of selects in parallel uses index information generated for one or more of the plurality of tokens to generate the one or more selects pointing to the one or more of the uncompressed symbols in the combined history window.

34. (Original) The method of claim 25, wherein the combined history window includes an uncompressed plurality of symbols from one or more previous decompression cycles and data bytes from the current decompression cycle, wherein said generating the plurality of selects in parallel comprises:

generating a first select to point to a data byte in the combined history window in response to a first token indicating that uncompressed data represented by the first token is the data byte; and

generating a second select to point to a first symbol in the combined history window in response to a second token indicating that uncompressed data represented by the second token includes the first symbol in the combined history window.

35. (Original) The method of claim 34, wherein the uncompressed data represented by the second token includes one or more symbols following the first symbol in the combined history window, wherein selects are generated to point to each of the one or more symbols in the combined history window comprising the uncompressed data represented by the second token.

36. (Original) A method for performing parallel decompression of compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, the method comprising:

receiving the compressed data, wherein the compressed data comprises tokens each describing one or more of the symbols in the uncompressed data;

examining a plurality of tokens in parallel from the compressed data in a current decompression cycle;

generating a plurality of selects in response to examining the one or more tokens, wherein each of the plurality of selects points to a symbol in a combined history window; and

generating the uncompressed data comprising the plurality of symbols using the plurality of selects.

37. (Original) The method of claim 36, wherein the plurality of selects are generated in parallel.

38. (Original) The method of claim 37, wherein the plurality of selects are generated substantially concurrently.

39. (Original) A method for performing parallel decompression of compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, the method comprising:

receiving the compressed data, wherein the compressed data comprises tokens each describing one or more of the symbols in the uncompressed data;

examining a plurality of tokens from the compressed data in parallel in a current decompression cycle;

generating a plurality of selects in parallel in response to examining the plurality of tokens in parallel, wherein zero or more of the plurality of selects point to a symbol in a history window, wherein remaining ones of the plurality of selects point to a current data byte from the current decompression cycle;

generating an uncompressed data stream comprising the plurality of symbols using the plurality of selects; and

storing the uncompressed plurality of symbols from the current decompression cycle in the history window.

40. (Original) A method for performing parallel decompression of compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, the method comprising:

receiving the compressed data, wherein the compressed data comprises tokens each describing one or more of the symbols in the uncompressed data;

examining a plurality of tokens from the compressed data in parallel in a current decompression cycle;

generating, for each token, size and count information and at least one of a data byte or index information, wherein a size for a token defines the number of bits comprising the token, wherein a count for a token defines the number of symbols in the uncompressed data described by the token;

generating a plurality of selects in parallel using the size and count information and at least one of the data byte or index information, wherein each select points to a symbol in a combined history window;

generating the uncompressed data stream comprising the plurality of symbols using the plurality of selects; and

storing the uncompressed plurality of symbols from the current decompression cycle in the combined history window.

41. (Original) The method of claim 40,

wherein the combined history window includes uncompressed plurality of symbols from zero or more previous decompression cycles and one or more data bytes from the current decompression cycle;

wherein, if data byte information is generated for a token, said generating the plurality of selects in parallel uses the data byte information generated for the token to generate a select to one of the one or more data bytes in the combined history window; and

wherein, if index information is generated for the token, said generating the plurality of selects in parallel uses the index information generated for the token to generate one or more selects to one or more of the uncompressed plurality of symbols in the combined history window.

42. (Original) A method for performing parallel decompression of compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, wherein a plurality of decoders are operable to decode in parallel a plurality of tokens from the compressed data in a current decompression cycle, wherein, for a current decompression cycle, the method comprises:

a) examining a section of the compressed data, wherein the section of the compressed data comprises tokens each describing one or more of the symbols in the uncompressed data;

b) determining if there is a decoder available in the plurality of decoders;

c) extracting a token from the section of the compressed data in response to determining that there is a decoder available; and

d) decompressing the token, wherein decompressing the token comprises:

generating one or more selects from the token, wherein the one or more selects point to one or more symbols in a combined history window;

generating the uncompressed data stream comprising the one or more symbols using the one or more selects; and

storing the uncompressed one or more symbols generated from the token in the combined history window.

43. (Original) The method of claim 42, wherein b) and c) are repeated in the current decompression cycle until there are no more decoders available in the plurality of decoders.

44. (Original) The method of claim 42, wherein the plurality of tokens are extracted from the section of the compressed data by repeating b) and c) in the current decompression cycle, and wherein d) is performed in parallel for the plurality of tokens in the current decompression cycle.

45. (Original) The method of claim 44, repeating a) - d) for each of a plurality of decompression cycles until all of the tokens in the compressed data have been decompressed.

46. (Original) The method of claim 42, wherein each of the plurality of decoders is operable to decode one of the plurality of tokens in the current decompression cycle, and wherein c) further comprises:

assigning the extracted token to the available decoder for decoding; and

generating size and count information and at least one of a data byte or index information for the token being decoded on the available decoder.

47. (Original) The method of claim 46, wherein said generating the one or more selects for the token uses the size and count information and at least one of the data byte or index information;

wherein a size for a token defines the number of bits comprising the token; and

wherein a count for a token defines the number of symbols in the uncompressed data described by the token.

48. (Original) The method of claim 42,

wherein the combined history window includes one or more data bytes from the current decompression cycle; and

wherein one or more of the selects in the current decompression cycle point to one or more of the data bytes in the combined history window.

49. (Original) The method of claim 48, wherein said generating the one or more selects for the token uses index information generated for the token to generate a select pointing to one of the one or more data bytes in the combined history window.

50. (Original) The method of claim 42,

wherein the combined history window includes one or more uncompressed symbols from one or more previous decompression cycles; and

wherein one or more of the selects in the current decompression cycle point to one or more of the uncompressed symbols in the combined history window from the one or more previous decompression cycles.

51. (Original) The method of claim 50, wherein said generating the one or more selects in parallel uses index information generated for the token to generate the one or more selects pointing to the one or more of the uncompressed symbols in the combined history window.

52. (Original) A method for performing parallel decompression of compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, the method comprising:

receiving the compressed data, wherein the compressed data comprises tokens each describing one or more of the symbols in the uncompressed data;

examining a plurality of tokens from the compressed data in parallel in a current decompression cycle; and

generating the uncompressed data comprising the plurality of symbols in response to said examining.

53. (Original) The method of claim 52, further comprising:

generating a plurality of selects in response to examining the plurality of tokens in parallel, wherein each of the plurality of selects points to a symbol in a combined history window;

wherein said generating the uncompressed data uses the plurality of selects.

54. (Original) The method of claim 53, wherein the plurality of selects are generated in parallel.

55. (Currently amended) The method of claim ~~52~~53, wherein said generating the uncompressed data includes storing the uncompressed plurality of symbols from the current decompression cycle in the combined history window.

56. (Original) A system for performing parallel decompression of compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, wherein the uncompressed data comprises uncompressed symbols, the system comprising:

an input for receiving the compressed data, wherein the compressed data comprises tokens each describing one or more of the symbols in the uncompressed data;

a decompression engine for decompressing a plurality of tokens from the compressed data in a parallel fashion, wherein the decompression engine is operable to generate a plurality of uncompressed symbols described by the plurality of tokens; and

an output coupled to the decompression engine for outputting the plurality of uncompressed symbols in the uncompressed data in response to the decompression of the plurality of tokens.

57. (Original) The system of claim 56, wherein the decompression engine includes a combined history window comprising a plurality of uncompressed symbols, and wherein the decompression engine is further operable to:

a) examine the plurality of tokens from the compressed data in parallel in a current decompression cycle;

b) generate a plurality of selects in response to examining the plurality of tokens in parallel, wherein each of the plurality of selects points to an uncompressed symbol in the combined history window;

c) generate the uncompressed data comprising the uncompressed symbols pointed to by the plurality of selects; and

d) store the uncompressed symbols from the current decompression cycle in the combined history window.

58. (Original) The system of claim 56, wherein the decompression engine includes a combined history window comprising a plurality of uncompressed symbols, and wherein the decompression engine is further operable to:

a) examine one or more tokens from the compressed data in a current decompression cycle;

b) generate a plurality of selects in parallel in response to examining the one or more tokens in parallel, wherein each of the plurality of selects points to an uncompressed symbol in the combined history window;

c) generate the uncompressed data comprising the uncompressed symbols pointed to by the plurality of selects; and

d) store the uncompressed symbols from the current decompression cycle in the combined history window.

59. (Original) The system of claim 56, wherein the decompression engine includes a combined history window comprising a plurality of uncompressed symbols, and wherein the decompression engine is further operable to:

a) examine a plurality of tokens from the compressed data in parallel in a current decompression cycle;

b) generate a plurality of selects in parallel in response to examining the plurality of tokens in parallel, wherein each of the plurality of selects points to an uncompressed symbol in the combined history window;

c) generate the uncompressed data comprising the uncompressed symbols pointed to by the plurality of selects; and

d) store the uncompressed symbols from the current decompression cycle in the combined history window.

60. (Original) The system of claim 59, wherein the decompression engine is further operable to repeat a) through d) until all of the tokens in the compressed data have been decompressed.

61. (Original) The system of claim 59,

wherein, in the current decompression cycle prior to said storing the uncompressed plurality of symbols, the combined history window includes an uncompressed plurality of symbols from any previous decompression cycles and zero or more data bytes from the current decompression cycle.

62. (Original) The system of claim 59, wherein the decompression engine is further operable to:

generate, for each token, size and count information and at least one of a data byte or index information, wherein a size for a token defines the number of bits comprising the token, wherein a count for a token defines the number of symbols in the uncompressed data described by the token;

wherein said generating the plurality of selects in parallel uses the size and count information and at least one of the data byte or index information for each of the plurality of tokens.

63. (Original) The system of claim 59,

wherein the combined history window includes one or more data bytes from the current decompression cycle; and

wherein said generating the plurality of selects in parallel uses data byte information generated for the token to generate a select to one of the one or more data bytes in the combined history window.

64. (Original) The system of claim 59,

wherein the combined history window includes the uncompressed plurality of symbols from one or more previous decompression cycles; and

wherein said generating the plurality of selects in parallel uses index information generated for the token to generate one or more selects to one or more of the uncompressed plurality of symbols from one or more previous decompression cycles in the combined history window.

65. (Original) The system of claim 59, wherein the combined history window includes the uncompressed plurality of symbols from one or more previous decompression cycles and data bytes from the current decompression cycle, and wherein said generating the plurality of selects in parallel comprises:

generating a first select to point to a data byte in the combined history window in response to a first token indicating that uncompressed data represented by the first token is the data byte; and

generating a second select to point to a first symbol in the combined history window in response to a second token indicating that uncompressed data represented by the second token includes the first symbol in the combined history window.

66. (Original) The system of claim 65, wherein the uncompressed data represented by the second token includes one or more symbols following the first symbol in the combined history window, wherein selects are generated to point to each of the one or more symbols in the combined history window comprising the uncompressed data represented by the second token.

67. (Original) The system of claim 59, wherein the combined history window includes an uncompressed plurality of symbols from one or more previous decompression cycles and data bytes

from the current decompression cycle, wherein said generating the plurality of selects in parallel comprises:

generating a first select to point to a first symbol being decompressed from a first token in the current decompression cycle, wherein the first select is generated in response to a second token indicating that uncompressed data represented by the second token includes the first symbol, and wherein the first symbol is not in the combined history window.

68. (Original) The system of claim 67, further comprising resolving the first select to point to one of a symbol in the current combined history window or a data byte in the current combined history window.

69. (Original) The system of claim 67, further comprising copying a second select being generated for the first token to the first select, wherein the second select points to one of a symbol in the combined history window or a data byte in the combined history window.

70. (Original) The system of claim 59, wherein storing the uncompressed plurality of symbols from the current decompression cycle in the combined history window comprises removing from the combined history window at least a portion of an uncompressed plurality of symbols from one or more previous decompression cycles.

71. (Original) The system of claim 59, wherein said receiving the compressed data, said examining a plurality of tokens, said generating a plurality of selects, and said generating the uncompressed data comprising the plurality of symbols are performed substantially concurrently in a pipelined fashion.

72. (Original) A system for performing parallel decompression of compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, wherein the uncompressed data comprises uncompressed symbols, the system comprising:

an input for receiving the compressed data, wherein the compressed data comprises tokens each describing one or more of the symbols in the uncompressed data;

a combined history window comprising a plurality of uncompressed symbols;

a first stage operable to examine a plurality of tokens from the compressed data in parallel in a current decompression cycle;

a second stage operable to generate a plurality of selects in parallel in response to the first stage examining the plurality of tokens in parallel, wherein each of the plurality of selects points to an uncompressed symbol in the combined history window;

a third stage operable to:

generate the uncompressed data comprising the uncompressed symbols pointed to by the plurality of selects generated by the second stage; and

store the uncompressed symbols from the current decompression cycle in the combined history window; and

an output coupled to the third stage for outputting the plurality of uncompressed symbols in the uncompressed data in response to the decompression of the plurality of tokens.

73. (Original) The system of claim 72,

wherein, in the current decompression cycle prior to said storing the uncompressed plurality of symbols, the combined history window includes an uncompressed plurality of symbols from zero or more previous decompression cycles and zero or more data bytes from the current decompression cycle.

74. (Original) The system of claim 72, wherein the first stage is further operable to:

generate, for each token, size and count information and at least one of a data byte or index information;

wherein the second stage generating the plurality of selects in parallel uses the size and count information and at least one of the data byte or index information for each of the plurality of tokens.

75. (Original) The system of claim 72, wherein the first stage includes a plurality of decoders, wherein each decoder is operable to receive one token from the plurality of tokens in the current decompression cycle, and wherein a decoder is operable to:

generate, for a token received by the decoder, size and count information and at least one of a data byte or index information;

wherein the second stage generating the plurality of selects in parallel uses the size and count information and at least one of the data byte or index information generated by the decoder.

76. (Original) The system of claim 72, wherein the first stage is further operable to:

extract a portion of the compressed data as an input data, wherein the input data includes the plurality of tokens; and

extract one or more tokens from the input data.

77. (Original) The system of claim 76, wherein the first stage includes a plurality of decoders, wherein each decoder is operable to receive a token from the plurality of tokens in the current decompression cycle, wherein, during said extracting the one or more tokens, the first stage is further operable to:

a) determine if there are more tokens to be uncompressed in the input data;

b) determine if there is a decoder available in the plurality of decoders;

c) extract a token from the input data to be a current token in response to determining that a decoder is available;

d) determine the number of uncompressed symbols to be generated by the current token;

e) assign the current token to the available decoder; and

f) repeat a) through e) until a terminating condition is encountered.

78. (Original) The system of claim 77, wherein the terminating condition is encountered when:

step a) determines there are no more tokens in the input data; or

step b) determines there are no decoders available in the plurality of decoders; or

step d) determines that a total number of uncompressed symbols to be generated from the plurality of tokens to be examined in parallel has met or exceeded a maximum output width.

79. (Original) The system of claim 77, wherein, during said extracting the one or more tokens, the first stage is further operable to:

determine if a token in the input data is a complete token or an incomplete token, wherein the token is extracted from the input data to be the current token in response to determining the token is a complete token;

wherein the terminating condition is encountered in response to determining the token is an incomplete token.

80. (Original) The system of claim 76, wherein the first stage is further operable to:

shift the compressed data by a total size of one or more decompressed tokens, where shifting the compressed data prepares a next input data to be extracted.

81. (Original) The system of claim 71, wherein the first, second and third stages operate substantially concurrently in a pipelined fashion.

82. (Original) The system of claim 71, further comprising:

a first pipe register coupled to the first and second stages; and

a second pipe register coupled to the second and third stages;

wherein the pipe registers are operable to:

a) receive from a previous stage data generated in a first decompression cycle;

b) store the data generated in the first decompression cycle;

c) send the data generated in the first decompression cycle to the next stage for processing by the next stage in the first decompression cycle.

83. (Original) A system for performing parallel decompression of compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, wherein the uncompressed data comprises uncompressed symbols, the system comprising:

an input for receiving the compressed data, wherein the compressed data comprises tokens each describing one or more of the symbols in the uncompressed data;

a combined history window comprising a plurality of uncompressed symbols;

a first stage operable to examine a plurality of tokens from the compressed data in parallel in a current decompression cycle;

a second stage operable to generate a plurality of preliminary selects in parallel in response to the first stage examining the plurality of tokens in parallel, wherein each of the plurality of preliminary selects points to an uncompressed symbol in the combined history window or to another of the preliminary selects in the current decompression cycle;

a third stage operable to generate a plurality of final selects in parallel in response to the second stage generating the plurality of preliminary selects, wherein each of the plurality of final selects points to an uncompressed symbol in the combined history window;

a fourth stage operable to:

generate the uncompressed data comprising the uncompressed symbols pointed to by the plurality of final selects generated by the third stage; and

store the uncompressed symbols from the current decompression cycle in the combined history window; and

an output coupled to the fourth stage for outputting the plurality of uncompressed symbols in the uncompressed data in response to the decompression of the plurality of tokens.

84. (Original) The system of claim 83,

wherein, in the current decompression cycle prior to said storing the uncompressed plurality of symbols, the combined history window includes an uncompressed plurality of symbols from zero or more previous decompression cycles and zero or more data bytes from the current decompression cycle.

85. (Original) The system of claim 83, wherein the first stage is further operable to:

generate, for each token, size and count information and at least one of a data byte or index information;

wherein the second stage generating the plurality of preliminary selects in parallel uses the size and count information and at least one of the data byte or index information for each of the plurality of tokens.

86. (Original) The system of claim 83, wherein the first stage includes a plurality of decoders, wherein the decoders are operable to each receive one token from the plurality of tokens in the current decompression cycle, and wherein the decoders are operable to:

generate, for a token received by the decoder, size and count information and at least one of a data byte or index information;

wherein the second stage generating the plurality of preliminary selects in parallel uses the size and count information and at least one of the data byte or index information generated by the decoder.

87. (Original) The system of claim 83, wherein a preliminary select generated by the second stage includes select information and overflow information, and wherein the third stage uses the select information and overflow information to generate a final select.

88. (Original) The system of claim 83, wherein the first stage is further operable to:

extract a portion of the compressed data as an input data, wherein the input data includes the plurality of tokens; and

extract one or more tokens from the input data.

89. (Original) The system of claim 88, wherein the first stage includes a plurality of decoders, wherein each decoder is operable to receive a token from the plurality of tokens in the current decompression cycle, wherein, during said extracting the one or more tokens, the first stage is further operable to:

a) determine if there are more tokens to be uncompressed in the input data;

b) determine if there is a decoder available in the plurality of decoders;

c) extract a token from the input data to be a current token in response to determining that a decoder is available;

d) determine the number of uncompressed symbols to be generated by the current token;

e) assign the current token to the available decoder; and

f) repeat a) through e) until a terminating condition is encountered.

90. (Original) The system of claim 89, wherein the terminating condition is encountered when:

step a) determines there are no more tokens in the input data; or

step b) determines there are no decoders available in the plurality of decoders; or

step d) determines that a total number of uncompressed symbols to be generated from the plurality of tokens to be examined in parallel has met or exceeded a maximum output width.

91. (Original) The system of claim 89, wherein, during said extracting the token, the first stage is further operable to:

determine if the token in the input data is a complete token or an incomplete token, wherein the token is extracted from the input data to be the current token in response to determining the token is a complete token;

wherein the terminating condition is encountered in response to determining the token is an incomplete token.

92. (Original) The system of claim 88, wherein the first stage is further operable to:

shift the compressed data by a total size of one or more decompressed tokens, where shifting the compressed data prepares a next input data to be extracted.

93. (Original) The system of claim 83, wherein the first, second, third and fourth stages operate substantially concurrently in a pipelined fashion.

94. (Original) The system of claim 83, further comprising:

a first pipe register coupled between the first and second stages;

a second pipe register coupled between the second and third stages; and

a third pipe register coupled between the third and fourth stages;

wherein the pipe registers are operable to:

a) receive from a previous stage data generated in a first decompression cycle;

b) store the data generated in the first decompression cycle;

c) send the data generated in the first decompression cycle to the next stage for processing by the next stage in the first decompression cycle.

95. (Original) A memory controller, comprising:

memory control logic for controlling a memory;

a parallel decompression engine for decompressing compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, wherein the parallel decompression engine is operable to:

receive the compressed data, wherein the compressed data comprises tokens each describing one or more uncompressed symbols;

examine a plurality of tokens from the compressed data in parallel in a current decompression cycle;

generate a plurality of selects in parallel in response to examining the plurality of tokens in parallel, wherein each of the plurality of selects points to a symbol in a combined history window; and

generate the uncompressed data comprising the plurality of symbols using the plurality of selects.

96. (Original) The memory controller of claim 95, wherein the parallel decompression engine is further operable to:

store the uncompressed plurality of symbols from the current decompression cycle in the combined history window.

97. (Original) The memory controller of claim 95, wherein said examining the plurality of tokens includes generating, for each token, size and count information and at least one of a data byte or index information; and

wherein said generating the plurality of selects in parallel uses the size and count information and at least one of the data byte or index information for each of the plurality of tokens.

98. (Original) The memory controller of claim 95, wherein the combined history window includes an uncompressed plurality of symbols from one or more previous decompression cycles and data bytes from the current decompression cycle, wherein said generating the plurality of selects in parallel comprises:

generating a first select to point to a data byte in the combined history window in response to a first token indicating that uncompressed data represented by the first token is the data byte; and

generating a second select to point to a first symbol in the combined history window in response to a second token indicating that uncompressed data represented by the second token includes the first symbol in the combined history window.

99. (Original) The memory controller of claim 95, wherein the combined history window includes an uncompressed plurality of symbols from one or more previous decompression cycles and data bytes from the current decompression cycle, wherein said generating the plurality of selects in parallel comprises:

generating a first select to point to a first symbol being decompressed from a first token in the current decompression cycle, wherein the first select is generated in response to a second token indicating that uncompressed data represented by the second token includes the first symbol, and wherein the first symbol is not in the combined history window.

100. (Original) The memory controller of claim 99, further comprising copying a second select being generated for the first token to the first select, wherein the second select points to one of a symbol in the combined history window or a data byte in the combined history window.

101. (Original) A memory controller, comprising:

memory control logic for controlling a memory;

a parallel decompression engine for decompressing compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, wherein the parallel decompression engine comprises:

an input for receiving the compressed data, wherein the compressed data comprises tokens each describing one or more uncompressed symbols;

a combined history window comprising a plurality of uncompressed symbols;

token examination logic operable to examine a plurality of tokens from the compressed data in parallel in a current decompression cycle;

preliminary select generation logic operable to generate a plurality of preliminary selects in parallel in response to the token examination logic examining the plurality of tokens in parallel, wherein each of the plurality of preliminary selects points to an uncompressed symbol in the combined history window or to another of the preliminary selects in the current decompression cycle;

final select generation logic operable to generate a plurality of final selects in parallel in response to the preliminary select generation logic generating the plurality of preliminary selects, wherein each of the plurality of final selects points to an uncompressed symbol in the combined history window;

uncompressed data output logic operable to:

generate the uncompressed data comprising the uncompressed symbols pointed to by the plurality of final selects generated by the second stage; and

store the uncompressed symbols from the current decompression cycle in the combined history window; and

an output coupled to the uncompressed data output logic for outputting the plurality of uncompressed symbols in the uncompressed data in response to the decompression of the plurality of tokens.

102. (Original) The memory controller of claim 101, wherein the token examination logic includes a plurality of decoders, wherein the decoders are operable to each receive one token from the plurality of tokens in the current decompression cycle, and wherein the decoders are operable to:

generate, for each token, size and count information and at least one of a data byte or index information;

wherein the preliminary select generation logic uses the size and count information and at least one of the data byte or index information generated by the decoders to generate the plurality of preliminary selects in parallel.

103. (Original) A memory module, comprising:

one or more memory devices for storing data;

a parallel decompression engine for decompressing compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, wherein the parallel decompression engine is operable to:

receive the compressed data, wherein the compressed data comprises tokens each describing one or more uncompressed symbols;

examine a plurality of tokens from the compressed data in parallel in a current decompression cycle;

generate a plurality of selects in parallel in response to examining the plurality of tokens in parallel, wherein each of the plurality of selects points to a symbol in a combined history window;

generate uncompressed data comprising the plurality of symbols.

104. (Original) The memory module of claim 103, wherein the parallel decompression engine is further operable to:

store the uncompressed plurality of symbols from the current decompression cycle in the combined history window.

105. (Original) The memory module of claim 103, wherein said examining the plurality of tokens includes generating, for each token, size and count information and at least one of a data byte or index information; and

wherein said generating the plurality of selects in parallel uses the size and count information and at least one of the data byte or index information for each of the plurality of tokens.

106. (Original) The memory module of claim 103, wherein the combined history window includes an uncompressed plurality of symbols from one or more previous decompression cycles and data bytes from the current decompression cycle, wherein said generating the plurality of selects in parallel comprises:

generating a first select to point to a data byte in the combined history window in response to a first token indicating that uncompressed data represented by the first token is the data byte; and

generating a second select to point to a first symbol in the combined history window in response to a second token indicating that uncompressed data represented by the second token includes the first symbol in the combined history window.

107. (Original) The memory module of claim 103, wherein the combined history window includes an uncompressed plurality of symbols from one or more previous decompression cycles and data bytes from the current decompression cycle, wherein said generating the plurality of selects in parallel comprises:

generating a first select to point to a first symbol being decompressed from a first token in the current decompression cycle, wherein the first select is generated in response to a second token indicating that uncompressed data represented by the second token includes the first symbol, and wherein the first symbol is not in the combined history window.

108. (Original) The memory module of claim 103, further comprising copying a second select being generated for the first token to the first select, wherein the second select points to one of a symbol in the combined history window or a data byte in the combined history window.

109. (Original) A memory module, comprising:

one or more memory devices for storing data;

a parallel decompression engine for decompressing compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, wherein the parallel decompression engine comprises:

an input for receiving the compressed data, wherein the compressed data comprises tokens each describing one or more uncompressed symbols;

a combined history window comprising a plurality of uncompressed symbols;

token examination logic operable to examine a plurality of tokens from the compressed data in parallel in a current decompression cycle;

preliminary select generation logic operable to generate a plurality of preliminary selects in parallel in response to the token examination logic examining the plurality of tokens in parallel, wherein each of the plurality of preliminary selects points to an uncompressed symbol in the combined history window or to another of the preliminary selects in the current decompression cycle;

final select generation logic operable to generate a plurality of final selects in parallel in response to the preliminary select generation logic generating the plurality of preliminary selects, wherein each of the plurality of final selects points to an uncompressed symbol in the combined history window;

uncompressed data output logic operable to:

generate the uncompressed data comprising the uncompressed symbols pointed to by the plurality of final selects generated by the second stage; and

store the uncompressed symbols from the current decompression cycle in the combined history window; and

an output coupled to the uncompressed data output logic for outputting the plurality of uncompressed symbols in the uncompressed data in response to the decompression of the plurality of tokens.

110. (Original) The memory module of claim 109, wherein the token examination logic includes a plurality of decoders, wherein the decoders are operable to each receive one token from the plurality of tokens in the current decompression cycle, and wherein the decoders are operable to:

generate, for each token, size and count information and at least one of a data byte or index information;

wherein the preliminary select generation logic uses the size and count information and at least one of the data byte or index information generated by the decoders to generate the plurality of preliminary selects in parallel.

111. (Original) A computer system including a memory controller having an embedded parallel decompression engine for decompressing compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, the computer system comprising:

a CPU;

system memory which stores data used by said CPU for executing one or more applications;

a memory controller coupled to the system memory and the CPU, wherein the memory controller performs memory control functions for the system memory, wherein the memory controller includes the parallel decompression engine for decompressing compressed data transferred to or from the system memory;

wherein the parallel decompression engine is operable to:

receive the compressed data, wherein the compressed data comprises tokens each describing one or more uncompressed symbols;

examine a plurality of tokens from the compressed data in parallel in a current decompression cycle;

generate a plurality of selects in parallel in response to examining the plurality of tokens in parallel, wherein each of the plurality of selects points to a symbol in a combined history window;

generate uncompressed data comprising the plurality of symbols; and

store the uncompressed plurality of symbols from the current decompression cycle in the combined history window.

112. (Original) A computer system including a memory controller having an embedded parallel decompression engine for decompressing compressed data, wherein the compressed data comprises a compressed representation of uncompressed data, the uncompressed data having a plurality of symbols, the computer system comprising:

a CPU;

system memory which stores data used by said CPU for executing one or more applications;

a memory controller coupled to the system memory and the CPU, wherein the memory controller performs memory control functions for the system memory, wherein the memory controller includes the parallel decompression engine for decompressing compressed data transferred to or from the system memory;

wherein the parallel decompression engine comprises:

an input for receiving the compressed data, wherein the compressed data comprises tokens each describing one or more uncompressed symbols;

a combined history window comprising a plurality of uncompressed symbols;

token examination logic operable to examine a plurality of tokens from the compressed data in parallel in a current decompression cycle;

preliminary select generation logic operable to generate a plurality of preliminary selects in parallel in response to the token examination logic examining the plurality of tokens in parallel, wherein each of the plurality of preliminary selects points to an uncompressed symbol in the combined history window or to another of the preliminary selects in the current decompression cycle;

final select generation logic operable to generate a plurality of final selects in parallel in response to the preliminary select generation logic generating the plurality of preliminary selects, wherein each of the plurality of final selects points to an uncompressed symbol in the combined history window;

uncompressed data output logic operable to:

generate the uncompressed data comprising the uncompressed symbols pointed to by the plurality of final selects; and

store the uncompressed symbols from the current decompression cycle in the combined history window; and

an output coupled to the uncompressed data output logic for outputting the plurality of uncompressed symbols in the uncompressed data in response to the decompression of the plurality of tokens.